Amendments to the Claims

The enclosed is responsive to the Office Action mailed on June 27, 2007. At the time the Office Action was mailed claims 1-23 were pending. By way of the present response Applicant has: 1) amended no claims; 2) added no claims; and 3) canceled no claims. As such, claims 1-23 are now pending. A listing of claims follows:

In the Claims

1. (Original) A machine readable medium having stored thereon a plurality of control signals and when accessed by a processor, causing said processor to:

access a first set of bits as representing an instruction of a first plurality of instructions, said first plurality of instructions including a packed data instruction, a scalar floating point instruction, and a transition instruction to be executed between said packed data instruction and said scalar floating point instruction;

if said first set of bits represents the packed data instruction then generate a first set of control signals to cause said processor to execute the packed data instruction on packed data contents of a storage representing a programmer visible register file, wherein said storage representing the programmer visible register file is operated as a flat register file while executing said packed data instruction;

if said first set of bits represents the scalar floating point instruction then

generate the first set of control signals to cause said processor to execute the scalar floating point instruction on floating point contents of the storage representing the programmer visible register file, wherein said programmer visible register file is operated as a stack while executing said scalar floating point instruction; and

if said first set of bits represents the transition instruction then generate the first set of control signals to cause said processor to alter a tag data to indicate that the stack of the programmer visible register file is empty responsive to executing said transition instruction between execution of said packed data instruction and said scalar floating point instruction.

2. (Original) The machine readable medium Claim 1, when accessed by said processor, further causing the processor to:

generate the first set of control signals to cause said processor to alter a sign and exponent data to indicate that a floating point contents of the programmer visible register file represent not-a-number (NAN) if said first set of bits represents the packed data instruction.

3. (Original) The machine readable medium Claim 1, when accessed by said processor, further causing the processor to:

generate the first set of control signals to cause said processor to alter a top of stack data to indicate that top of the stack of the programmer visible register file is zero if said first set of bits represents the transition instruction or if said first set of bits represents said packed data instruction.

4. (Original) The machine readable medium Claim 1 wherein the plurality of control signals stored thereon are to be accessed by a CISC processor.

5. (Original) The machine readable medium Claim 1 wherein the plurality of control signals stored thereon are to be accessed by a RISC processor.

6. (Original) The machine readable medium Claim 1 wherein the plurality of control signals stored thereon are to be accessed by a VLIW processor.

7. (Original) The machine readable medium of Claim 6, when accessed by said processor, further causing the processor to:

access a second set of bits as representing a floating point state save instruction or a floating point environment save instruction;

generate a second set of control signals to cause said processor to store into memory a status data of the programmer visible register file comprising said tag data, said tag data indicating that the stack of the programmer visible register file is empty responsive to executing said floating point state save instruction or said floating point environment save instruction following execution of said transition instruction, and said tag data indicating that the stack of the programmer visible register file is not empty responsive to executing said floating point state save instruction or floating point environment save instruction following execution of said packed data instruction.

8. (Original) A computer-implemented method comprising:

accessing a first set of bits as representing a packed data instruction;

storing a first corresponding set of control bits to cause a processor to alter a top of stack data to zero and to operate on packed data contents of a storage representing a programmer visible register file as a flat register file;

accessing a second set of bits as representing a scalar floating point instruction;

storing a second corresponding set of control bits to cause the processor to operate on floating point data contents of the storage representing the programmer visible register file as a stack; and

accessing a third set of bits as representing a transition instruction to be executed between said packed data instruction and said scalar floating point instruction;

storing a third corresponding set of control bits to cause the processor to alter a tag data to indicate that the stack of the programmer visible register file is empty.

9. (Original) The computer-implemented method of Claim 8 wherein the first set of control bits further causes said processor to alter a sign and exponent data of the programmer visible register file to indicate not-a-number (NAN).

10. (Original) The computer-implemented method of Claim 9 wherein said processor is a CISC processor.

11. (Original) The computer-implemented method of Claim 9 wherein said processor is a RISC processor.

12. (Original) The computer-implemented method of Claim 9 wherein said processor is a VLIW processor.

13. (Original) The computer-implemented method of Claim 9 wherein the third set of control bits further causes said processor to alter the top of stack data to indicate that top of the stack of the programmer visible register file is zero.

14. (Original) The computer-implemented method of Claim 9 further comprising:

accessing a fourth set of bits as representing a floating point state save instruction or a floating point environment save instruction to be executed after said transition instruction;

storing a fourth corresponding set of control bits to cause the processor to store a status data comprising said tag data of the programmer visible register file, wherein said stored tag data is to indicate that the stack of the programmer visible register file is empty.

15. (Original) The computer-implemented method of Claim 9 further comprising:

accessing a fourth set of bits as representing a floating point state save instruction or a floating point environment save instruction to be executed after said packed data instruction;

storing a fourth corresponding set of control bits to cause the processor to store a status data comprising said tag data of the programmer visible

register file, wherein said stored tag data is to indicate that the stack of the programmer visible register file is not empty.

16. (Original) A computer system comprising:

an processor execution unit to execute control signals;

a bus coupled with said processor execution unit configurable to access one or more machine readable medium;

a first machine readable medium configurable to store a first plurality of instructions, said first plurality of instructions including a packed data instruction, a scalar floating point instruction, and a transition instruction to be executed between said packed data instruction and said scalar floating point instruction;

a second machine readable medium configurable to store a programmer visible register file and related environment information, said environment information including a tag data and a top of stack data for accessing said programmer visible register file in a stack referenced manner; and

a third machine readable medium having stored thereon a plurality of control signals that, when accessed by the processor execution unit, cause said processor execution unit to

access a first set of bits as representing an instruction of the first plurality of instructions from the first machine readable medium, to

generate a first set of control signals to cause said processor execution unit to alter the top of stack data to zero and operate on contents of the programmer visible register file as a flat register file when said first set of

bits represents the packed data instruction, to

generate the first set of control signals to cause said processor execution unit to operate on contents of the programmer visible register file in a stack referenced manner when said first set of bits represents the scalar floating point instruction, and to

generate the first set of control signals to cause said processor execution unit to alter the tag data to indicate that the stack of the programmer visible register file is empty when said first set of bits represents the transition instruction.

17. (Original) The computer system of Claim 16, wherein the plurality of control signals of the third machine readable medium being accessed by said processor execution unit, further causes the processor execution unit to:

access a second set of bits as representing a floating point state save instruction or a floating point environment save instruction from the first machine readable medium, to

generate a second set of control signals to cause said processor execution unit to store said environment information, the tag data indicating that the stack of the programmer visible register file is empty responsive to executing said floating point state save instruction or said floating point environment save instruction following execution of said transition instruction, or otherwise said tag data indicating that the stack of the programmer visible register file is not empty responsive to executing said floating point state save

instruction or floating point environment save instruction following execution of said packed data instruction.

18. (Original) The computer system of Claim 16 wherein the plurality of control signals of the third machine readable medium are to be accessed by a CISC processor execution unit.

19. (Original) The computer system of Claim 17 the third machine readable medium comprising micro code control signals.

20. (Original) The computer system of Claim 16 wherein the plurality of control signals of the third machine readable medium are to be accessed by a RISC processor.

21. (Original) The computer system of Claim 16 the plurality of control signals of the third machine readable medium comprising VLIW control signals to be accessed by a VLIW processor.

22. (Original) The computer system of Claim 20 the third machine readable medium comprising a read-only memory (ROM).

23. (Original) The computer system of Claim 21 the first machine readable medium comprising a dynamic random-access memory (DRAM).